PAT-NO: JP408272756A

DOCUMENT-IDENTIFIER: JP 08272756 A

TITLE: METHOD FOR STARTING MULTIPROCESSOR SYSTEM

PUBN-DATE: October 18, 1996

INVENTOR-INFORMATION:
NAME
YAMAGAMI, NOBUHIKO
NAKADA, YASUMASA
UETOKO, KATSUKI

ASSIGNEE-INFORMATION:

NAME	COUNTRY
TOSHIBA CORP	N/A

APPL-NO: JP07076291

APPL-DATE: March 31, 1995

INT-CL (IPC): G06F015/177, G06F001/24 , G06F011/16

ABSTRACT:

PURPOSE: To improve the reliability of the boot process of a multiprocessor system.

CONSTITUTION: The multiprocessor system which includes one boot ROM 2 storing codes for booting the system and allows plural processors 4, 5, and 6 to execute the codes stored in the boot ROM 2 is equipped with a system bus arbitrating mechanism 3 which selects one of processors outputting requests to acquire a system bus according to a rule prescribing the priority of the processors and gives system bus use permission at the request to acquire the bus outputted from all operable processors to fetch the codes in the boot ROM 2 when system resetting is reset, and does not give the system bus use permission to other processors until the one selected processor completes the execution of the codes stored in the boot ROM 2.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平8－272756

(43)公開日　平成８年(1996)10月18日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　15/177			Ｇ０６Ｆ　15/16	４２０Ｓ	
1/24			11/16	３１０Ｚ	
11/16	３１０		1/00	３５０Ｂ	

審査請求　未請求　請求項の数３　ＯＬ　（全 12 頁）

(21)出願番号　特願平7－76291

(22)出願日　平成７年(1995)３月31日

(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地

(72)発明者　山上　宜彦
東京都青梅市末広町２丁目９番地　株式会社東芝青梅工場内

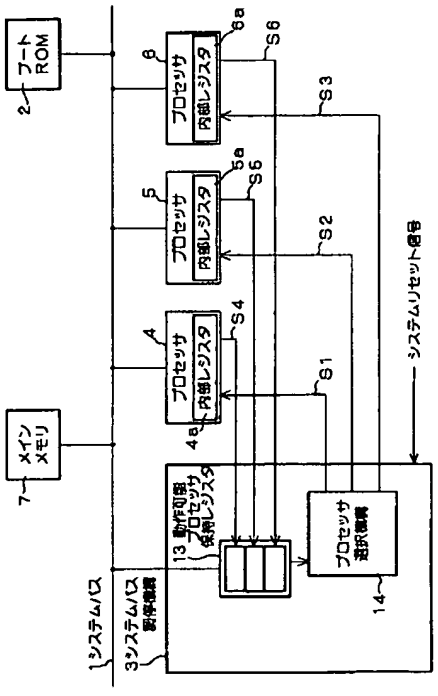
(72)発明者　中田　恭正
東京都青梅市末広町２丁目９番地　株式会社東芝青梅工場内

(72)発明者　上床　克樹
東京都青梅市末広町２丁目９番地　株式会社東芝青梅工場内

(74)代理人　弁理士　鈴江　武彦

(54)【発明の名称】　マルチプロセッサシステムの起動方法

(57)【要約】
【目的】マルチプロセッサシステムのブートプロセスの際の信頼性を向上させることを可能にする。
【構成】システムをブートするためのコードが入ったブートＲＯＭ２が１つ存在し、複数のプロセッサ４，５，６がブートＲＯＭ２に収められていたコードを実行するマルチプロセッサシステムにおいて、システムリセットを解除したときに、動作可能な全てのプロセッサから、ブートＲＯＭ２中のコードをフェッチするために出力されるシステムバス獲得要求に対して、システムバス獲得要求を出力したプロセッサの中からプロセッサの優先順位を規定する規則に従って１つを選択してシステムバス使用許可を与え、その他のプロセッサには選択された１つのプロセッサによってブートＲＯＭ２に記憶されたコードの実行が完了するまでシステムバス使用許可を与えないシステムバス調停機構３を具備して構成する。



1

【特許請求の範囲】

【請求項１】　システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、

システムリセットが解除されたとき、前記システムバスの獲得要求を出力したプロセッサの中から予め規定された優先度に基づいて１つのプロセッサを選択してシステムバスの使用許可を与え、選択されなかったその他のプロセッサには、前記選択されたプロセッサが前記ブートストラップの実行が完了するまで、システムバスの使用許可を与えないことを特徴とするマルチプロセッサシステムの起動方法。

【請求項２】　システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、

システムリセットが解除されたとき、前記複数のプロセッサからそれぞれ出力される前記システムバスの獲得要求に対して、順次システムバスの使用許可を与え、前記複数のプロセッサのそれぞれにより前記記憶装置に記憶されたブートストラップを実行させ、１つのプロセッサが前記ブートストラップの内のプロセッサの初期化ルーチンを終了した際、そのプロセッサに前記ブートストラップの実行を継続させるとともに、その他のプロセッサには、前記ブートストラップの内のプロセッサの初期化ルーチンを終了した後、以後のルーチンを実行させないことを特徴とするマルチプロセッサシステムの起動方法。

【請求項３】　システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、

システムリセットが解除されたとき、前記複数のプロセッサの中から予め規定された優先度に基づいて１つのプロセッサを選択して前記記憶装置に記憶されたブートストラップを実行させ、この選択されたプロセッサが所定時間内に前記ブートストラップの内のプロセッサの初期化ルーチンを終了した場合には、そのプロセッサに前記ブートストラップの実行を継続させ、この選択されたプロセッサが所定の時間内に前記ブートストラップの内のプロセッサの初期化ルーチンを終了しなかった場合には、前記優先度に基づいてその他のプロセッサを選択して前記記憶装置に記憶されたブートストラップを実行させることを特徴とするマルチプロセッサシステムの起動方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、マルチプロセッサシステムにおける起動方法に関する。

2

【０００２】

【従来の技術】従来、マルチプロセッサシステムにおいては、複数のプロセッサによって共用されるシステムバスの使用を、システムバス調停機構（アービター）によって制御している。

【０００３】システムリセットを解除する場合には、全てのプロセッサに対するリセット信号が解除される。その結果、動作可能な全てのプロセッサは、システムを起動させるためのブートプロセスを実行するためのコード（プログラム）、すなわち、ブートストラップが格納されたブートＲＯＭの先頭番地の命令をフェッチするため、システムバス獲得要求をシステムバス調停機構に出力する。

【０００４】システムバス調停機構は、全てのプロセッサに対して、優先順位を規定するある規則に従って、システムバス獲得要求の単位でシステムバス使用許可を送出する。この場合、システムバス使用許可を受信したプロセッサは、システムバスを獲得して処理を実行するが、結果的に、全てのプロセッサがブートＲＯＭの中に格納されているプログラムのある一定の処理（例えばプロセッサ自分自身を初期化する処理）まで実行する。

【０００５】そして、ある一定の処理以降では、１つのプロセッサを除いて、無意味なダミー処理を実行する無限ループに入り、ＯＳ（オペレーティングシステム）によってマルチプロセッサ化されるのを待つ。また、１つのプロセッサは、その後のブートＲＯＭの中に格納されているプログラムを実行する。この処理には、ボード上の周辺回路などの初期化、Ｉ／Ｏデバイスの初期化、ＯＳのローディングなどが含まれる。なお、ブートＲＯＭ中に格納されたプログラムを継続して実行する１つのプロセッサは、予め決められている優先度に基づいて、最も優先度が高いスロットに装着されているボードのプロセッサに決定される。この優先度は具体的には、例えば、プロセッサが搭載されたボードを装置本体に装着するためのスロットに設定しておく。

【０００６】

【発明が解決しようとする課題】このように従来のマルチプロセッサシステムでは、ブートＲＯＭの中に格納されているプログラムを完全に継続して（ボード上の周辺回路などの初期化、Ｉ／Ｏデバイスの初期化、ＯＳのローディングなどまで）実行する１つのプロセッサが予め決められていた。

【０００７】このため、その決められたプロセッサが故障していると、ブートプロセスが実行できず、システムが立ち上がらなくなってしまうという問題があった。すなわち、１つのプロセッサの故障だけで、他に正常なプロセッサが存在しているとしても、システム全体が使用できなくなってしまう場合があるため、ブートプロセスの信頼性を、より向上させることが望まれていた。

【０００８】本発明は前記のような事情を考慮してなさ

れたもので、マルチプロセッサシステムのブートプロセスの際の信頼性を向上させることが可能なマルチプロセッサシステムの起動方法を提供することを目的とする。

【0009】

【課題を解決するための手段】本発明は、システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、システムリセットが解除されたとき、前記システムバスの獲得要求を出力したプロセッサの中から予め規定された優先度に基づいて１つのプロセッサを選択してシステムバスの使用許可を与え、選択されなかったその他のプロセッサには、前記選択されたプロセッサが前記ブートストラップの実行が完了するまで、システムバスの使用許可を与えないことを特徴とする。

【0010】また、システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、システムリセットが解除されたとき、前記複数のプロセッサからそれぞれ出力される前記システムバスの獲得要求に対して、順次システムバスの使用許可を与え、前記複数のプロセッサのそれぞれにより前記記憶装置に記憶されたブートストラップを実行させ、１つのプロセッサが前記ブートストラップの内のプロセッサの初期化ルーチンを終了した際、そのプロセッサに前記ブートストラップの実行を継続させるとともに、その他のプロセッサには、前記ブートストラップの内のプロセッサの初期化ルーチンを終了した後、以後のルーチンを実行させないことを特徴とする。

【0011】また、システムを起動するためのブートストラップを記憶した記憶装置を持ち、システムバスを介して前記記憶装置と接続された複数のプロセッサを有するマルチプロセッサシステムにおいて、システムリセットが解除されたとき、前記複数のプロセッサの中から予め規定された優先度に基づいて１つのプロセッサを選択して前記記憶装置に記憶されたブートストラップを実行させ、この選択されたプロセッサが所定時間内に前記ブートストラップの内のプロセッサの初期化ルーチンを終了した場合には、そのプロセッサに前記ブートストラップの実行を継続させ、この選択されたプロセッサが所定の時間内に前記ブートストラップの内のプロセッサの初期化ルーチンを終了しなかった場合には、前記優先度に基づいてその他のプロセッサを選択して前記記憶装置に記憶されたブートストラップを実行させることを特徴とする。

【0012】

【作用】このような構成によれば、システムリセットが解除された際、ブートプロセスを実行するためにシステムバス獲得要求を出すことができる場合には、そのプロセッサは正常であるものとし、正常なプロセッサから１つを選択してブートプロセスを実行させる。従って、故障したプロセッサが存在し、そのプロセッサのブートプロセスを実行させる優先順位が最も高いとしても、正常なプロセッサが存在すれば、ブートプロセスを確実に実行させることができる。また、ブートプロセスの段階で正常と判別されたプロセッサを、マルチプロセッサ化の対象とすることができる。

【0013】また、システムリセットが解除された際、各プロセッサに対してブートプロセスを実行させ、ブートプロセスとして最初に実行されるプロセッサ自身を初期化するルーチンが終了できる場合には、そのプロセッサは正常であるものとし、正常なプロセッサから１つを選択して（最初に初期化するルーチンを完了したプロセッサ）、ブートプロセスをそれ以降も継続して実行させる。従って、システムバス獲得要求を出力できたものの、プロセッサ自身を初期化するルーチンが実行できない故障があるプロセッサが存在し、そのプロセッサのブートプロセスを実行させる優先順位が最も高いとしても、正常なプロセッサが存在すれば、ブートプロセスを確実に実行させることができる。

【0014】また、同様にしてプロセッサ自身を初期化するルーチンを完了できるか否かによって故障を判別するが、ブートプロセスを実行するプロセッサの優先順位を規定する規則に従って１つのプロセッサを選択し、正常なプロセッサが得られるまで、順次、対象とするプロセッサを変更してブートプロセスを試行する。従って、１つでも正常なプロセッサが存在すれば、ブートプロセスを実行させることができる。

【0015】

【実施例】以下、図面を参照して本発明の実施例を説明する。図１は本発明の第１実施例に係わるマルチプロセッサシステムの概略構成を示すブロック図である。図１に示すように、第１実施例のマルチプロセッサシステムは、システムバス１を介して、ブートＲＯＭ２、システムバス調停機構３、複数（本実施例では３つ）のプロセッサ４，５，６、及びメインメモリ７が相互に接続されて構成されている。

【0016】システムバス１は、プロセッサ４，５，６、ブートＲＯＭ２、メインメモリ７、システムバス調停機構３などの間のデータのやり取りを行なう。ブートＲＯＭ２は、ブートプロセスに必要なコード（プログラム）を格納するためのもので、プロセッサ自身の初期化、ボード上の周辺回路などの初期化、Ｉ／Ｏデバイスの初期化、ＯＳのローディングなどの処理に関するプログラムが格納されている。

【0017】システムバス調停機構３は、プロセッサ４，５，６などからのシステムバス１に対する使用要求を調停するものである。システムバス調停機構３は、システムバス１と接続されている他、各プロセッサ４，

5，6との間でプロセッサ制御信号及びシステムバス調停機構制御信号の送受信を行なう。なお、プロセッサ制御信号は、システムバス調停機構3からプロセッサに対するリセット信号、システムバス使用許可信号を含んでいる。また、システムバス調停機構制御信号は、プロセッサからシステムバス調停機構3に対するシステムバス獲得要求信号を含んでいる。各信号が送受信される動作の詳細については後述する。

【0018】システムバス調停機構3には、動作可能プロセッサ保持レジスタ13、及びプロセッサ選択機構14が設けられている。動作可能プロセッサ保持レジスタ13は、システム内で動作可能なプロセッサを示す情報を各プロセッサ毎に保持するためのもので、システム内のプロセッサの数と同数のフィールドが設けられ、フィールドとプロセッサが対応づけられている。動作可能プロセッサ保持レジスタ13には、物理アドレスが割り当てられており、外部から（他の何れのプロセッサからでも）読み書きが可能となっている。プロセッサ選択機構14は、動作可能プロセッサ保持レジスタ13に格納された情報を参照して、プロセッサからのシステムバス獲得要求信号に対して、動作可能なプロセッサのみにシステムバス使用許可信号を送出、あるいはシステムリセット信号に応じてプロセッサにプロセッサ制御信号のリセット信号を送出するものである。

【0019】プロセッサ4，5，6は、システムバス獲得要求信号をシステムバス調停機構3に送出することによって、システムバス1の使用権を獲得し、命令等を実行する。なお、本実施例の前提条件として、全てのプロセッサ4，5，6が持つ状態レジスタ、プログラムカウンタなど、プロセッサ4，5，6がそれぞれ動作するために必要な内部レジスタ4a，5a，6aには物理アドレスが割り当てられており、外部から（他のどのプロセッサからも）書き込みが可能であるとする。

【0020】メインメモリ7は、プロセッサ4，5，6によって実行されるコード（プログラム）やデータ等を格納するためのもので、ブート時には、ブートROM2に格納された最初に実行されるプログラムによって、ブートROM2中のその他のブートプロセスのためのコードがブートROM2から読み出され格納される。また、メインメモリ7に格納されたブートプロセスのコードがプロセッサによって実行されて、OS（オペレーティングシステム）がローディングされる他、各種プログラムやデータ等が格納される。

【0021】次に、第1実施例の動作について、図2に示すフローチャートを参照しながら説明する。まず、システムバス調停機構3は、システムリセット信号がオンであるときには、プロセッサ選択機構14によって各プロセッサ4，5，6のそれぞれに対してプロセッサ制御信号S1，S2，S3のリセット信号を出力し、プロセッサにリセットをかけ続ける。その結果、プロセッサ

4、5、6は、命令フェッチなどの動作を全く行なわない。

【0022】システムリセット信号がオフとなりシステムリセットが解除されると、システムバス調停機構3は、プロセッサ制御信号S1，S2，S3のリセット信号を解除（オフ）することによって、プロセッサ4，5，6の動作を開始させる（ステップA1）。

【0023】この時、プロセッサ4，5，6は、ブートROM2の先頭番地の命令をフェッチするために、それぞれシステムバス調停機構制御信号S4，S5，S6のシステムバス獲得要求信号を用いて、システムバス調停機構3に対してシステムバス獲得要求を出す（ステップA2）。

【0024】システムバス調停機構3は、システムバス獲得要求を出したプロセッサが何れであるか、すなわちシステムバス獲得要求信号を正常に出力できた動作可能なプロセッサが何れであるかを動作可能プロセッサ保持レジスタ13に保持しておく（ステップA3）。

【0025】また、システムバス調停機構3は、プロセッサ選択機構14によって、動作可能プロセッサ保持レジスタ13に動作可能である情報が保持されたプロセッサの中から、優先順位を規定する規則に従って（プロセッサが搭載されたボードのスロットへの実装位置等）、最も優先度が高い1つのプロセッサを選択してシステムバス使用許可信号を出力する。また、システムバス調停機構3は、選択されなかった他のプロセッサに対してリセット信号を出力する（ステップA4）。

【0026】ここで、システムバス調停機構3からシステムバス使用許可信号を入力したプロセッサをプロセッサ4であるものとする。すなわち、プロセッサ選択機構14は、プロセッサ4にプロセッサ制御信号S1のシステムバス使用許可信号を用いてシステムバスの使用許可を与え、プロセッサ5，6にはプロセッサ制御信号S2，S3のリセット信号を用いて命令フェッチなどの動作を全く行なわないようにする。つまり、この状態では、動作するプロセッサはプロセッサ4だけとなる。

【0027】プロセッサ4は、システムバス1の使用権を獲得したので、システムバス1を介してコードを読み込んでブートプロセス、すなわちプロセッサ自身の初期化、ボード上の周辺回路などの初期化、I／Oデバイスの初期化などの処理を実行する（ステップA5）。

【0028】ブートプロセスが完了すると、プロセッサ4は、動作可能プロセッサ保持レジスタ13を読んで、動作可能なプロセッサが何れかを認識し、動作が可能なプロセッサに対して状態レジスタ、プログラムカウンタなどに適切な値を書き込んで、プロセッサ自身が初期化された状態にしてマルチプロセッサ化できる状態にする。また、システムバス調停機構3からのプロセッサ制御信号S2，S3のリセット信号を解除させて、プロセッサ5，6の動作を開始させる（ステップA6，A

7）。

【００２９】前述した第１実施例において、障害を持つプロセッサを排除できることを説明する。プロセッサ４，５，６の中に障害を持つプロセッサがある場合には、まず、システムリセットが解除された際に、システムバス調停機構３にシステムバス獲得要求信号を出すことができない。

【００３０】このため、障害があるためにシステムバス獲得要求信号が出力できなかったプロセッサについては、システムバス調停機構３の動作可能プロセッサ保持レジスタ１３に正常であることを示す情報が設定されない。

【００３１】従って、障害のあったプロセッサには、ブートプロセスを実行した１つのプロセッサにより、障害のあったプロセッサの状態レジスタ、プログラムカウンタなどの値が書き込まれず、そのプロセッサへのシステムバス調停機構３からのリセット信号が解除されないことになり、そのプロセッサを排除することができる。

【００３２】このようにして、リセット信号が解除された際に、システムバス調停機構３に対してシステムバス獲得要求信号を出力できない故障のあったプロセッサが排除され、正常な他のプロセッサによってブートプロセスを実行させることができる。従って、ブートプロセスを実行すべき優先度の最も高い１つのプロセッサに故障があっても、他に正常なプロセッサが存在すれば、システムを起動させることができる。

【００３３】次に、本発明の第２実施例について説明する。図３は本発明の第２実施例に係わるマルチプロセッサシステムの概略構成を示すブロック図である。図３に示すように、システムバス１を介して、ブートＲＯＭ２、システムバス調停機構２３、複数（本実施例では３つ）のプロセッサ４，５，６、及びメインメモリ７が相互に接続されて構成されている。なお、図１に示す第１実施例のマルチプロセッサシステムと同一構成部分については同一符号を付して説明を省略する。

【００３４】システムバス調停機構２３は、プロセッサ４，５，６などからのシステムバス１に対する使用要求を調停するものである。システムバス調停機構２３は、システムバス１と接続されている他、各プロセッサ４，５，６との間でプロセッサ制御信号及びシステムバス調停機構制御信号の送受信を行なう。なお、プロセッサ制御信号は、システムバス調停機構２３からプロセッサに対するリセット信号、システムバス使用許可信号を含んでいる。また、システムバス調停機構制御信号は、プロセッサからシステムバス調停機構２３に対するシステムバス獲得要求信号を含んでいる。各信号が送受信される動作の詳細については後述する。

【００３５】システムバス調停機構２３には、動作可能プロセッサ保持レジスタ３３、プロセッサ選択機構３４、及び初期化制御フラグ３５が設けられている。動作

可能プロセッサ保持レジスタ３３は、システム内で動作可能なプロセッサを示す情報を各プロセッサ毎に保持するためのもので、システム内のプロセッサの数と同数のフィールドが設けられ、フィールドとプロセッサが対応づけられている。動作可能プロセッサ保持レジスタ３３には、物理アドレスが割り当てられており、外部から（他の何れのプロセッサからでも）読み書きが可能となっている。プロセッサ選択機構３４は、動作可能プロセッサ保持レジスタ３３に格納された情報を参照して、プロセッサからのシステムバス獲得要求信号に対して、動作可能なプロセッサのみにシステムバス使用許可信号を送出、あるいはシステムリセット信号に応じてプロセッサにプロセッサ制御信号のリセット信号を送出するものである。初期化制御フラグ３５は、複数のプロセッサの中で１つのプロセッサにブートプロセスを継続して実行させるために用いられる。初期化制御フラグ３５は、物理アドレスが割り当てられており、外部から（他の何れのプロセッサからでも）読み書きが可能となっている。

【００３６】次に、第２実施例の動作について、図４に示すフローチャートを参照しながら説明する。まず、システムバス調停機構２３は、システムリセット信号がオンであるときには、プロセッサ選択機構３４によって各プロセッサ４，５，６のそれぞれに対してプロセッサ制御信号Ｓ１，Ｓ２，Ｓ３のリセット信号を出力し、プロセッサにリセットをかけ続ける。その結果、プロセッサ４、５、６は、命令フェッチなどの動作を全く行なわない。

【００３７】システムリセット信号がオフとなりシステムリセットが解除されると、システムバス調停機構２３は、プロセッサ制御信号Ｓ１，Ｓ２，Ｓ３のリセット信号を解除（オフ）することによって、プロセッサ４，５，６の動作を開始させる（ステップＢ１）。

【００３８】この時、プロセッサ４，５，６は、ブートＲＯＭ２の先頭番地の命令をフェッチするために、それぞれシステムバス調停機構制御信号Ｓ４，Ｓ５，Ｓ６のシステムバス獲得要求信号を用いて、システムバス調停機構２３に対してシステムバス獲得要求を出す（ステップＢ２）。

【００３９】システムバス調停機構２３は、各プロセッサからのシステムバス獲得要求信号に対して、１つのプロセッサにシステムバス使用許可信号を出力し、システムバス１の使用権を与える（ステップＢ３）。

【００４０】システムバス使用許可信号を入力したプロセッサは、ブートＲＯＭ２のプログラム（コード）を読み込み、ブートプロセスの実行を開始する。ブートプロセスでは、まずプロセッサ自身を初期化する処理（初期化ルーチン）が実行される（ステップＢ４）。

【００４１】システムバス調停機構２３は、順次、システムバス獲得要求信号を出力したプロセッサにシステムバス使用許可を与える。このため、各プロセッサは、そ

れぞれプロセッサ自身を初期化する処理を実行する。

【0042】この結果、プロセッサ4，5，6の何れか1つが、最初にブートROM2に格納されたコード（プログラム）のプロセッサ自身を初期化するルーチンの実行を完了する（ステップB5）。ここでは、プロセッサ4が、最初に初期化ルーチンを完了したものとする。ブートROM2には、プロセッサ自身を初期化するルーチンが完了した後に、次のような処理を実行させるコードが格納されている。

【0043】すなわち、初期化ルーチンを最初に完了したプロセッサ4は、システムバス調停機構23の動作可能プロセッサ保持レジスタ33の自プロセッサに対応するフィールドに、初期化ルーチンが完了して正常であることを示すようにフラグを立てる（ステップB6）。

【0044】そして、プロセッサ4は、初期化制御フラグ35を参照し、フラグが立っていなければ、既に初期化ルーチンを完了したプロセッサが存在していることを示すようにフラグを立てる（ステップB7，B8）。プロセッサ4は、プロセッサ自身の初期化ルーチンに続くブートプロセス、すなわちボード上の周辺回路などの初期化、I／Oデバイスの初期化など、それ以降のコードの実行を続ける（ステップB9）。

【0045】一方、プロセッサ5，6は、初期化ルーチンを正常に完了できたら、プロセッサ4と同様にして、システムバス調停機構23の動作可能プロセッサ保持レジスタ33の自プロセッサに対応するフィールドにフラグを立てる（ステップB6）。

【0046】そして、プロセッサ5，6は、初期化制御フラグ35を参照するが、2番目移行にプロセッサ自身の初期化ルーチンを完了した場合、既にフラグが立っているので（ステップB7）、初期化ルーチンを継続して実行せず、所定の状況となるまで実行される無意味な処理、すなわち無限ループを実行する（ステップB10）。例えば、無限ループには、OSによってマルチプロセッサ化される際に、分岐命令に書き換えられる対象となるダミー命令（nop命令）が含まれている。

【0047】プロセッサ4は、正常にブートプロセスを実行するので、OSをローディングし実行する。プロセッサ4は、OSのマルチプロセッサ化するための処理に基づいて、システムバス調停機構23の動作可能プロセッサ保持レジスタ33を参照して、動作可能な（プロセッサ自身の初期化が完了した）プロセッサを判別する。

【0048】プロセッサ4は、動作可能なプロセッサ、すなわちプロセッサ5，6が実行しているメインメモリ7中に格納されている無限ループ中のダミー命令を、マルチプロセッサ処理を実行する処理へ分岐する分岐命令に書き換えて、無限ループの処理から出して実質的に動作を開始させる。

【0049】なお、プロセッサ自身を初期化するルーチンを完了した2番目以降のプロセッサに対して、無限ループの処理から出すためには、前述のような命令を書き換える方法の他に、無限ループによって所定回数以上の命令を実行した場合に、無限ループからマルチプロセッサ処理を実行する処理へ分岐するようになっていても良い。この場合、無限ループから出る前に、プロセッサ4によるブートプロセスが完了している必要がある。また、その他の方法であっても良い。

【0050】前述した第2実施例において、障害を持つプロセッサを排除できることを説明する。プロセッサ4，5，6の中に障害を持つプロセッサがある場合には、まず、そのプロセッサは、プロセッサ自身を初期化するルーチンを実行することができずに、動作可能プロセッサ保持レジスタ33に自プロセッサに対応するフィールドにフラグを立てる処理まで実行することができない。

【0051】従って、OSによってマルチプロセッサ化する際に、動作可能なプロセッサとは認識されないことになるので、そのプロセッサを排除することができる。このようにして、ブートROM2に格納されたプロセッサ自身を初期化するルーチンを実行できない故障のあったプロセッサが排除され、正常な他のプロセッサによってブートプロセスを実行させることができる。従って、ブートプロセスを実行すべき優先度の最も高い1つのプロセッサに故障があっても、他に正常なプロセッサが存在すれば、システムを起動させることができる。

【0052】次に、本発明の第3実施例について説明する。図5は本発明の第3実施例に係わるマルチプロセッサシステムの概略構成を示すブロック図である。図5に示すように、システムバス1を介して、ブートROM2、システムバス調停機構43、複数（本実施例では3つ）のプロセッサ4，5，6、及びメインメモリ7が相互に接続されて構成されている。なお、図1に示す第1実施例のマルチプロセッサシステムと同一構成部分については同一符号を付して説明を省略する。

【0053】システムバス調停機構43は、プロセッサ4，5，6などからのシステムバス1に対する使用要求を調停するものである。システムバス調停機構43は、システムバス1と接続されている他、各プロセッサ4，5，6との間でプロセッサ制御信号及びシステムバス調停機構制御信号の送受信を行なう。なお、プロセッサ制御信号は、システムバス調停機構43からプロセッサに対するリセット信号、システムバス使用許可信号を含んでいる。また、システムバス調停機構制御信号は、プロセッサからシステムバス調停機構43に対するシステムバス獲得要求信号を含んでいる。各信号が送受信される動作の詳細については後述する。

【0054】システムバス調停機構43には、動作可能プロセッサ保持レジスタ53、プロセッサ制御部54、及びカウンタ55が設けられている。動作可能プロセッサ保持レジスタ53は、システム内で動作可能なプロセ

ッサを示す情報を各プロセッサ毎に保持するためのもので、システム内のプロセッサの数と同数のフィールドが設けられ、フィールドとプロセッサが対応づけられている。動作可能プロセッサ保持レジスタ５３には、物理アドレスが割り当てられており、外部から（他の何れのプロセッサからでも）読み書きが可能となっている。プロセッサ制御部５４は、プロセッサからのシステムバス獲得要求信号に対して、動作可能なプロセッサにシステムバス使用許可信号を送出、あるいはシステムリセット信号に応じてプロセッサにプロセッサ制御信号のリセット信号を送出するものである。また、プロセッサ制御部５４は、動作可能プロセッサ保持レジスタ５３に格納された情報、及びカウンタ５５からの通知に基づいて、動作可能なプロセッサによってブートプロセスを継続して実行させるものである。カウンタ５５は、プロセッサがブートプロセスを開始してから所定の時間を計測するためのもので、プロセッサがブートＲＯＭ２内のプロセッサ自身を初期化するルーチンの実行が終了するまでに要する時間に比べて十分に長い時間（例えば１．５倍～２倍程度）の計測を行なうものとする。例えば、カウンタ５５は、システムクロックに同期するカウンタで、プロセッサ自身を初期化するルーチンの実行に要するサイクル数よりも十分に長いサイクル数をカウントできるものとする。カウンタ５５は、所定の時間の計測を完了した場合にプロセッサ制御部５４に通知する。

【００５５】次に、第３実施例の動作について、図６に示すフローチャートを参照しながら説明する。まず、システムバス調停機構４３は、システムリセット信号がオンであるときには、プロセッサ制御部５４によって各プロセッサ４，５，６のそれぞれに対してプロセッサ制御信号Ｓ１，Ｓ２，Ｓ３のリセット信号を出力し、プロセッサにリセットをかけ続ける。その結果、プロセッサ４、５、６は、命令フェッチなどの動作を全く行なわない。

【００５６】システムリセット信号がオフとなりシステムリセットが解除されると、システムバス調停機構４３は、複数のプロセッサから優先順位を規定する規則に従って（プロセッサが搭載されたボードのスロットへの実装位置等）、最も優先度が高い１つのプロセッサを選択してリセット信号を解除（オフ）することによって動作を開始させる。システムバス調停機構４３は、その他のプロセッサに対してはリセット信号を出力したままにしておく（ステップＣ１）。

【００５７】ここで、システムバス調停機構４３からリセット信号が解除されたプロセッサをプロセッサ４であるものとする。すなわち、プロセッサ制御部５４は、プロセッサ４にプロセッサ制御信号Ｓ１のリセット信号を解除し、またシステムバス獲得要求信号に応じたシステムバス使用許可信号を用いてシステムバスの使用許可を与え、プロセッサ５，６にはプロセッサ制御信号Ｓ２，

Ｓ３のリセット信号を用いて命令フェッチなどの動作を全く行なわないようにしておく。

【００５８】プロセッサ４は、システムバス１の使用権を獲得したので、システムバス１を介してコードを読み込んでブートプロセスの動作を開始する。ブートプロセスでは、まずプロセッサ自身を初期化する処理が実行される（ステップＣ２）。

【００５９】一方、カウンタ５５は、ブートプロセスの動作が開始されると同時に、時間の計測を開始する（ステップＣ３）。プロセッサ制御部５４は、システムバス使用許可信号を出力したプロセッサ、すなわちブートプロセスのプロセッサ自身の初期化を実行しているプロセッサに対応する動作可能プロセッサ保持レジスタ５３のフィールドに、フラグが設定されるかを監視する（ステップＣ４）。

【００６０】ブートプロセスを実行するプロセッサ４は、ブートＲＯＭ２内のコード（プログラム）のうち、プロセッサ自身を初期化するルーチンを終了すると、動作可能プロセッサ保持レジスタ５３の自プロセッサ４に対応するフィールドに、正常であることを示すようにフラグを立てる（ステップＣ５）。

【００６１】プロセッサ制御部５４は、現在動作しているプロセッサ４のフィールドにフラグが設定されたことを検出すると、カウンタ５５をリセットして、時間の計測を終了させる（ステップＣ６）。プロセッサ４は、そのままブートＲＯＭ２内のプロセッサ自身の初期化ルーチンに続くブートプロセス、すなわちボード上の周辺回路などの初期化、Ｉ／Ｏデバイスの初期化などの実行を継続する（ステップＣ７）。

【００６２】ブートプロセスを継続して実行させるプロセッサが得られると、プロセッサ制御部５４は、その他のプロセッサ（プロセッサ５，６）に対して、順次、システムバス使用許可信号によってシステムバス１の使用を許可しながら、ブートプロセスを実行させる（ステップＣ８）。

【００６３】２番目以降にブートプロセスを実行するプロセッサ５，６は、プロセッサ自身を初期化するルーチンを完了すると、システムバス調停機構４３の動作可能プロセッサ保持レジスタ５３の自プロセッサに対応するフィールドに、初期化ルーチンが完了して正常であることを示すようにフラグを立てる（ステップＣ９）。

【００６４】プロセッサ５，６は、動作可能プロセッサ保持レジスタ５３へのフラグの設定が完了すると、所定の状況となるまで実行される無意味な処理、すなわち無限ループを実行する。

【００６５】無限ループの処理を実行するプロセッサ５，６は、前述した第２実施例と同様にして、正常にブートプロセスを実行したプロセッサ４によってメインメモリ７にローディングされたＯＳによって、マルチプロセッサ化される際に、命令が書き換えられるなどにより

実質的に動作を開始する。

【0066】一方、動作可能プロセッサ保持レジスタのフラグ53へのフラグの設定が、何らかの要因によって遅れた場合、もしくは不可能になった場合、カウンタ55は、所定の時間の計測を完了し、その旨をプロセッサ制御部54に通知する（ステップC10）。

【0067】プロセッサ制御部54は、カウンタ55からの通知を受けると、その時に動作中のプロセッサ（この場合はプロセッサ4）に対し、故障等の異常があるものとしてブートプロセスの実行を停止させるためにリセット信号を出力する（ステップC11）。

【0068】次に、プロセッサ制御部54は、プロセッサ4の次に優先度が高い1つのプロセッサ（プロセッサ5とする）を選択してシステムバス使用許可信号を出力し（リセット解除）、前述したプロセッサ4と同様にしてブートプロセス、すなわちプロセッサ自身を初期化する処理を実行させる（ステップC1，C2）。

【0069】この際、カウンタ55は、ブートプロセスを実行させるプロセッサが変更されたことによって、時間の計測を初期状態から開始する（ステップC3）。以下、前述と同様にして処理する。

【0070】すなわち、プロセッサにブートプロセスを実行させて、プロセッサ自身を初期化するルーチンが完了できなかった場合、順次、対象とするプロセッサを変更し、システム内の全てのプロセッサについて試行するまで繰り返して行なう。

【0071】このようにして、ブートプロセスを実行すべき1つのプロセッサを選択して、プロセッサ自身を初期化するルーチンを試行させ、正常に初期化処理が完了すればそのままブートプロセスを継続して実行させる。そして、プロセッサ自身を初期化するルーチンが所定の時間内に完了できなかった場合には、次のプロセッサを

選択して、同様にブートプロセス（初期化ルーチン）を試行する。従って、システム内に1つでも動作可能なプロセッサが存在すれば、ブートROM2内のコード（プログラム）を、プロセッサ自身を初期化するルーチンの後も継続して実行させることができ、システムを起動させることができる。

【0072】

【発明の効果】以上詳述したように本発明によれば、複数のプロセッサの中で故障したプロセッサが存在しても、正常な他のプロセッサによってブートプロセスを実行し、システムを起動することができるので、信頼性の高いマルチプロセッサシステムを構成することが可能となるものである。

【図面の簡単な説明】

【図1】本発明の第1実施例に係わるマルチプロセッサシステムの構成を示すブロック図。

【図2】第1実施例の動作を説明するためのフローチャート。

【図3】本発明の第2実施例に係わるマルチプロセッサシステムの構成を示すブロック図。

【図4】第2実施例の動作を説明するためのフローチャート。

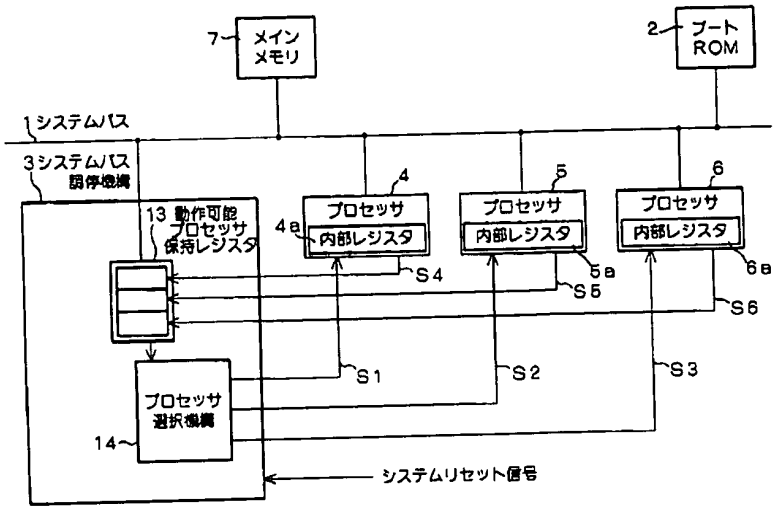
【図5】本発明の第3実施例に係わるマルチプロセッサシステムの構成を示すブロック図。

【図6】第3実施例の動作を説明するためのフローチャート。

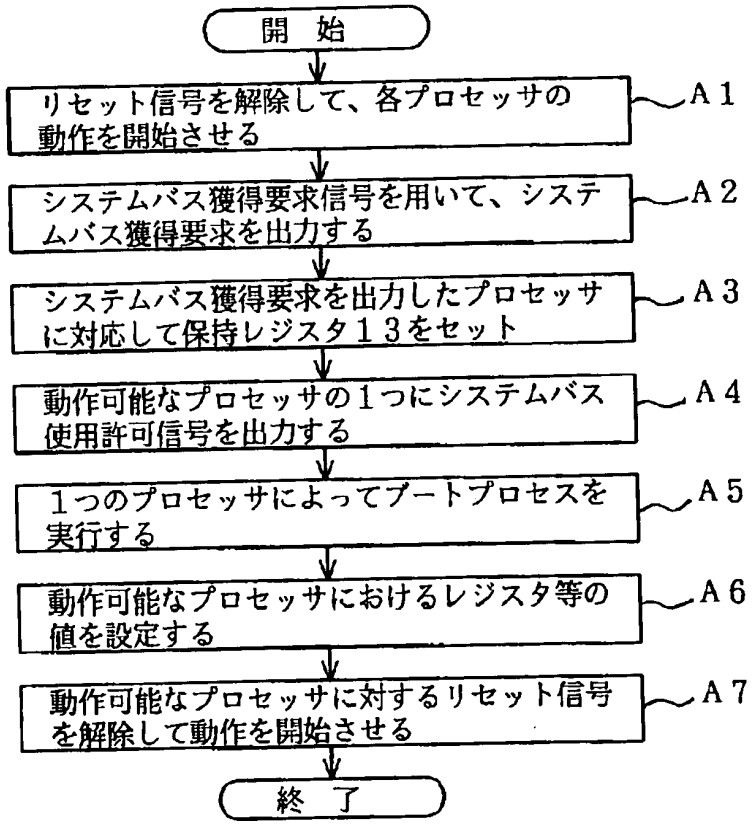
【符号の説明】

1…システムバス、2…ブートROM、3，23，43…システムバス調停機構、4，5，6…プロセッサ、7…メインメモリ、S1，S2，S3…プロセッサ制御信号、S4，S5，S6…システムバス調停機構制御信号。

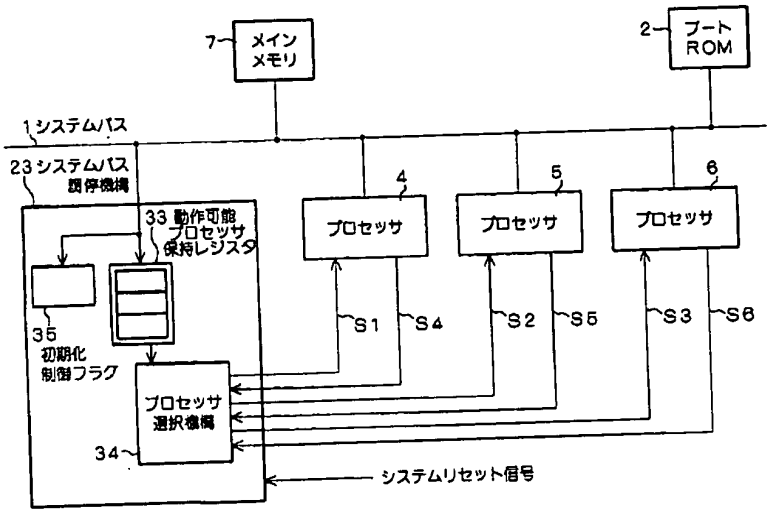
【図1】


7
メイン
メモリ
2
ブート
ROM
1システムバス
3システムバス
開停機構
13 動作可能
プロセッサ
保持レジスタ
4
プロセッサ
内部レジスタ
4a
5
プロセッサ
内部レジスタ
5a
6
プロセッサ
内部レジスタ
6a
S4
S5
S6
S1
S2
S3
プロセッサ
選択機構
14
システムリセット信号

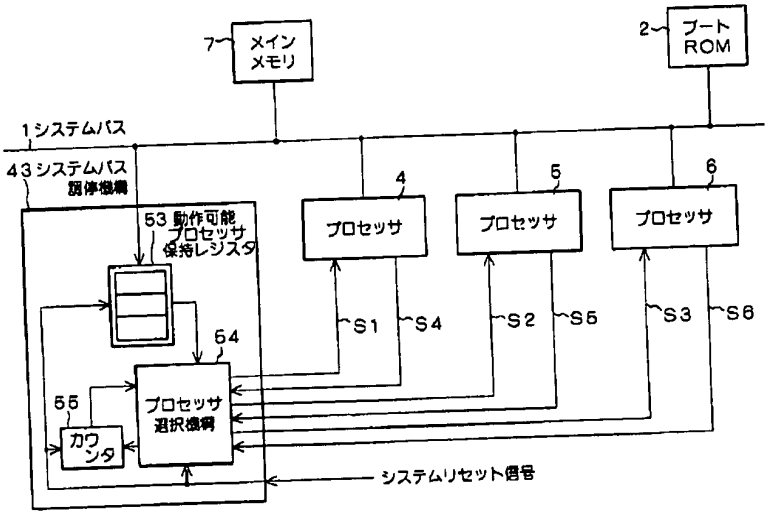
【図2】


開　始
リセット信号を解除して、各プロセッサの動作を開始させる
A1
システムバス獲得要求信号を用いて、システムバス獲得要求を出力する
A2
システムバス獲得要求を出力したプロセッサに対応して保持レジスタ１３をセット
A3
動作可能なプロセッサの１つにシステムバス使用許可信号を出力する
A4
１つのプロセッサによってブートプロセスを実行する
A5
動作可能なプロセッサにおけるレジスタ等の値を設定する
A6
動作可能なプロセッサに対するリセット信号を解除して動作を開始させる
A7
終　了

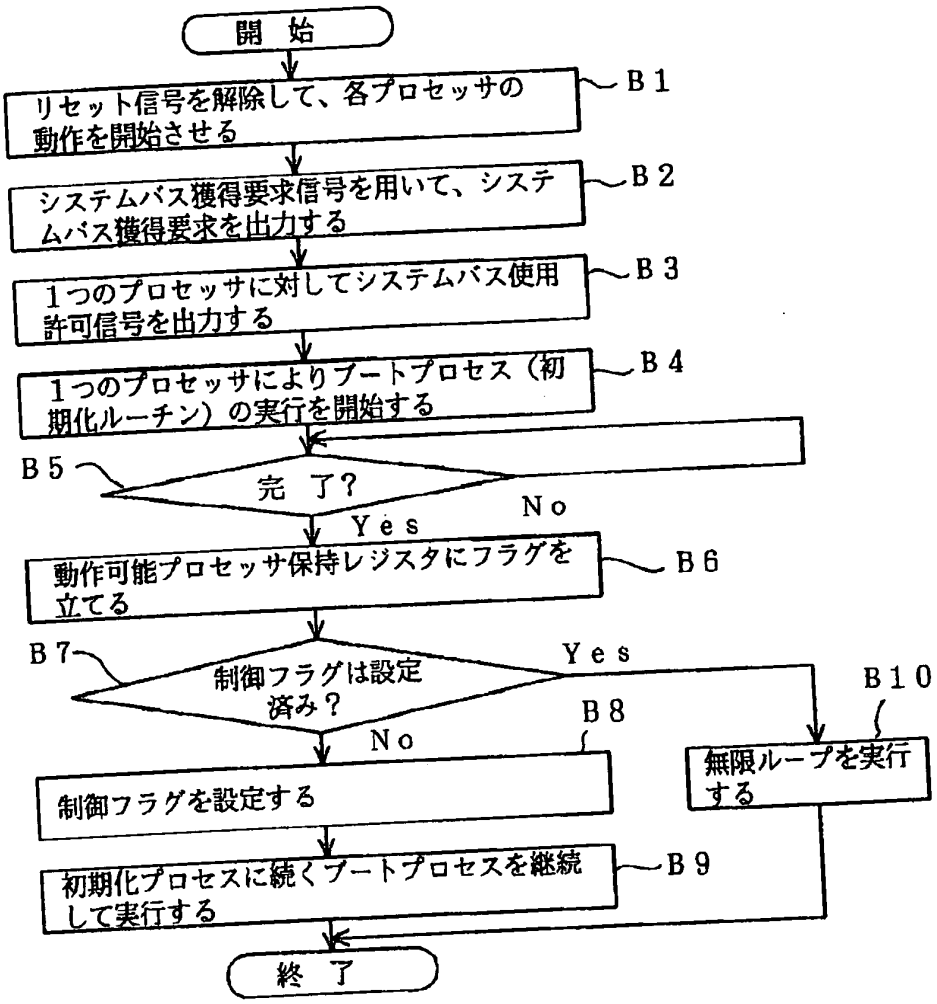
【図3】


7
メイン
メモリ
2
ブート
ROM
1システムバス
23システムバス
調停機構
33 動作可能
プロセッサ
保持レジスタ
35
初期化
制御フラグ
プロセッサ
選択機構
34
4
5
6
プロセッサ
プロセッサ
プロセッサ
S1
S4
S2
S5
S3
S6
システムリセット信号

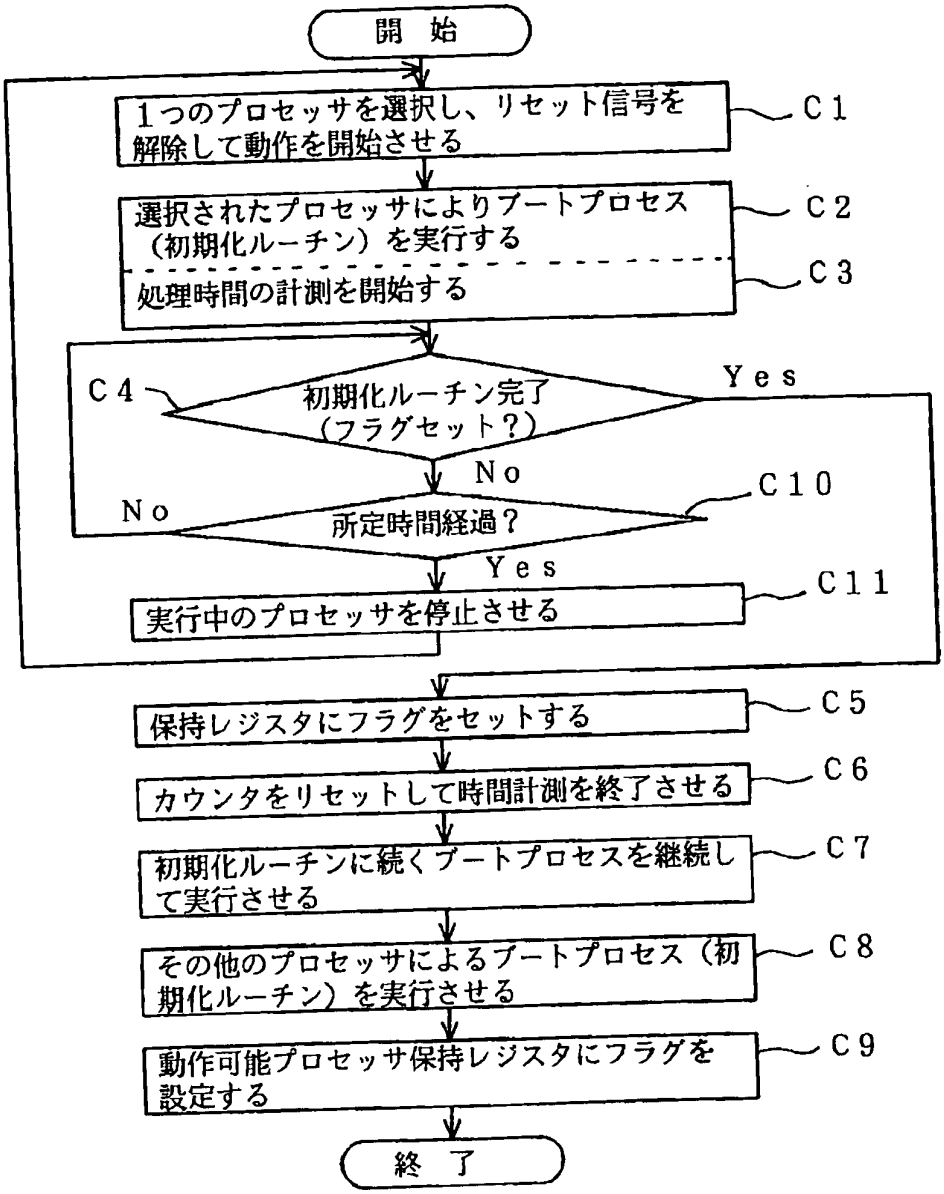
【図5】


7
メイン
メモリ
2
ブート
ROM
1システムバス
43システムバス
調停機構
53 動作可能
プロセッサ
保持レジスタ
54
55
カウンタ
プロセッサ
選択機構
4
5
6
プロセッサ
プロセッサ
プロセッサ
S1
S4
S2
S5
S3
S6
システムリセット信号

【図4】


開　始
リセット信号を解除して、各プロセッサの動作を開始させる
B1
システムバス獲得要求信号を用いて、システムバス獲得要求を出力する
B2
1つのプロセッサに対してシステムバス使用許可信号を出力する
B3
1つのプロセッサによりブートプロセス（初期化ルーチン）の実行を開始する
B4
B5
完　了?
No
Yes
動作可能プロセッサ保持レジスタにフラグを立てる
B6
B7
制御フラグは設定済み?
Yes
B10
B8
No
制御フラグを設定する
無限ループを実行する
初期化プロセスに続くブートプロセスを継続して実行する
B9
終　了

【図6】


開　始
1つのプロセッサを選択し、リセット信号を解除して動作を開始させる
C1
選択されたプロセッサによりブートプロセス（初期化ルーチン）を実行する
C2
処理時間の計測を開始する
C3
C4
初期化ルーチン完了（フラグセット？）
Yes
No
No
所定時間経過？
C10
Yes
実行中のプロセッサを停止させる
C11
保持レジスタにフラグをセットする
C5
カウンタをリセットして時間計測を終了させる
C6
初期化ルーチンに続くブートプロセスを継続して実行させる
C7
その他のプロセッサによるブートプロセス（初期化ルーチン）を実行させる
C8
動作可能プロセッサ保持レジスタにフラグを設定する
C9
終　了

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]
[Claim 1] In a multiprocessor system which has two or more processors which had the storage which memorized a bootstrap for starting a system, and were connected with said storage through a system bus When a system reset is canceled, based on a priority beforehand specified out of a processor which outputted an acquisition demand of said system bus, choose one processor, and licence of a system bus is given. A starting method of a multiprocessor system that said selected processor is characterized by not giving licence of a system bus at it until activation of said bootstrap is completed in a processor of others which were not chosen.
[Claim 2] In a multiprocessor system which has two or more processors which had the storage which memorized a bootstrap for starting a system, and were connected with said storage through a system bus As opposed to an acquisition demand of said system bus outputted from said two or more processors, respectively when a system reset is canceled A bootstrap which licence of a system bus was given one by one, and said two or more processors were alike, respectively, and was memorized more by said storage is performed. When one processor ends initialization routine of a processor of said bootstraps, while making the processor continue activation of said bootstrap A starting method of a multiprocessor system characterized by not performing future routines at them after ending initialization routine of a processor of said bootstraps in other processors.
[Claim 3] In a multiprocessor system which has two or more processors which had the storage which memorized a bootstrap for starting a system, and were connected with said storage through a system bus When a system reset is canceled, a bootstrap which chose one processor based on a priority beforehand specified out of said two or more processors, and was memorized by said storage is performed. When this selected processor ends initialization routine of a processor of said bootstraps in predetermined time When that processor is made to continue activation of said bootstrap and this selected processor does not end initialization routine of a processor of said bootstraps in predetermined time amount A starting method of a multiprocessor system characterized by performing a bootstrap which chose other processors based on said priority, and was memorized by said storage.

[Translation done.]

* NOTICES *

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Industrial Application] This invention relates to the starting method in a multiprocessor system.
[0002]
[Description of the Prior Art] Conventionally, in a multiprocessor system, use of the system bus shared by two or more processors is controlled by the system bus mediation device (arbiter).
[0003] When canceling a system reset, the reset signal to all processors is canceled. Consequently, since all the processors that can operate fetch the instruction of the code for performing the boot process for starting a system (program), i.e., the head address of a boot ROM with which the bootstrap was stored, they output a system bus acquisition demand to a system bus mediation device.
[0004] A system bus mediation device sends out system bus licence in the unit of a system bus acquisition demand to all processors according to a certain regulation which specifies priority. In this case, although the processor which received system bus licence gains a system bus and performs processing, all processors perform it as a result to the fixed processing (for example, processing which initializes itself) with the program stored in the boot ROM. [processor]
[0005] And henceforth [a certain fixed processing], it goes into the endless loop which performs meaningless dummy processing except for one processor, and waits to be multiprocessor-ized by OS (operating system). Moreover, one processor performs the program stored in the subsequent boot ROM. Initialization of the circumference circuit on a board etc., initialization of an I/O device, loading of OS, etc. are included in this processing. In addition, one processor which continues and performs the program stored in the boot ROM is determined as the processor of the board with which the slot with the highest priority is equipped based on the priority decided beforehand. This priority is set as the slot for specifically equipping the main part of equipment with the board on which the processor was carried.
[0006]
[Problem(s) to be Solved by the Invention] Thus, in the conventional multiprocessor system, one processor which continues completely and performs the program stored in the boot ROM (to initialization of the circumference circuit on a board etc., initialization of an I/O device, loading of OS, etc.) was decided beforehand.
[0007] For this reason, when that decided processor was out of order, a boot process could not be performed but there was a problem that a system would not start. That is, though the processor normal otherwise exists only by failure of one processor, since there was a case where it becomes impossible to use the whole system, to raise the reliability of a boot process more was desired.
[0008] This invention was made in consideration of the above situations, and aims at offering the starting method of the multiprocessor system which can raise the reliability in the case of the boot process of a multiprocessor system.
[0009]
[Means for Solving the Problem] In a multiprocessor system which has two or more processors which this invention had the storage which memorized a bootstrap for starting a system, and were connected

with said storage through a system bus When a system reset is canceled, based on a priority beforehand specified out of a processor which outputted an acquisition demand of said system bus, choose one processor, and licence of a system bus is given. Said selected processor is characterized by not giving licence of a system bus at a processor of others which were not chosen until activation of said bootstrap is completed.

[0010] Moreover, have the storage which memorized a bootstrap for starting a system, and it sets to a multiprocessor system which has two or more processors connected with said storage through a system bus. As opposed to an acquisition demand of said system bus outputted from said two or more processors, respectively when a system reset is canceled A bootstrap which licence of a system bus was given one by one, and said two or more processors were alike, respectively, and was memorized more by said storage is performed. When one processor ends initialization routine of a processor of said bootstraps, while making the processor continue activation of said bootstrap After ending initialization routine of a processor of said bootstraps, it is characterized by not performing future routines at other processors.

[0011] Moreover, have the storage which memorized a bootstrap for starting a system, and it sets to a multiprocessor system which has two or more processors connected with said storage through a system bus. When a system reset is canceled, a bootstrap which chose one processor based on a priority beforehand specified out of said two or more processors, and was memorized by said storage is performed. When this selected processor ends initialization routine of a processor of said bootstraps in predetermined time When that processor is made to continue activation of said bootstrap and this selected processor does not end initialization routine of a processor of said bootstraps in predetermined time amount It is characterized by performing a bootstrap which chose other processors based on said priority, and was memorized by said storage.

[0012]

[Function] When according to such a configuration a system reset is canceled and a system bus acquisition demand can be advanced in order to perform a boot process, the processor considers as a normal thing, chooses one from a normal processor, and performs a boot process. Therefore, though the priority which performs the boot process of the processor is the highest, if the broken processor exists, and a normal processor exists, a boot process can be performed certainly. Moreover, the processor distinguished as it is normal in the phase of a boot process can be set as the object of multiprocessor-izing.

[0013] Moreover, when a system reset is canceled and the routine which initializes the processor itself which is made to perform a boot process to each processor, and is first performed as a boot process can be ended, the processor considers as a normal thing, chooses one from a normal processor (processor which completed the routine initialized first), and it or subsequent ones continues and performs a boot process. Therefore, although the system bus acquisition demand has been outputted, a processor with the failure which cannot perform the routine which initializes the processor itself exists, and though the priority which performs the boot process of the processor is the highest, if a normal processor exists, a boot process can be performed certainly.

[0014] Moreover, although failure is distinguished by the ability of the routine which initializes the processor itself similarly to be completed, one by one, the target processor is changed and a boot process is tried until it chooses one processor according to the regulation which specifies the priority of the processor which performs a boot process and a normal processor is obtained. Therefore, a boot process can be performed if at least one normal processor exists.

[0015]

[Example] Hereafter, the example of this invention is explained with reference to a drawing. Drawing 1 is the block diagram showing the outline configuration of the multiprocessor system concerning the 1st example of this invention. As shown in drawing 1 , through the system bus 1, a boot ROM 2, the system bus mediation device 3, the processors 4, 5, and 6 of plurality (this example three), and main memory 7 are connected mutually, and the multiprocessor system of the 1st example is constituted.

[0016] A system bus 1 exchanges the data between processors 4, 5, and 6, a boot ROM 2, main memory

7, the system bus mediation device 3, etc. A boot ROM 2 is for storing a code (program) required for a boot process, and the program about processing of initialization of own initialization of a processor, the circumference circuit on a board, etc., initialization of an I/O device, loading of OS, etc. is stored.
[0017] The system bus mediation device 3 arbitrates the use demand to the system bus 1 from processors 4, 5, and 6 etc. It connects with the system bus 1, and also the system bus mediation device 3 performs transmission and reception of a processor control signal and a system bus mediation device control signal among each processors 4, 5, and 6. In addition, the processor control signal includes the reset signal to a processor, and the system bus licence signal from the system bus mediation device 3. Moreover, the system bus mediation device control signal includes the system bus acquisition demand signal over the system bus mediation device 3 from the processor. About the details of the actuation by which each signal is transmitted and received, it mentions later.
[0018] The processor holding register 13 which can be operated, and the processor optional feature 14 are formed in the system bus mediation device 3. The processor holding register 13 which can be operated is for holding the information which shows the processor which can operate within a system for every processor, the field of the number of the processors in a system and the same number is prepared, and the field and a processor are matched. The physical address is assigned to the processor holding register 13 which can be operated, and it can write from the exterior (from any of other processors). The processor optional feature 14 sends out the reset signal of a processor control signal for a system bus licence signal only to the processor which can operate to the system bus acquisition demand signal from a processor at a processor according to sending out or a system-reset signal with reference to the information stored in the processor holding register 13 which can be operated.
[0019] By sending out a system bus acquisition demand signal to the system bus mediation device 3, processors 4, 5, and 6 acquire the royalty of a system bus 1, and execute an instruction etc. In addition, as a prerequisite of this example, the physical address is assigned to the internal registers 4a, 5a, and 6a required in order that processors 4, 5, and 6 may operate, respectively, and that it can write in (every processor of other) carries out the status register which all the processors 4, 5, and 6 have, a program counter, etc. from the exterior.
[0020] Main memory 7 is for storing a code (program), data, etc. which are performed by processors 4, 5, and 6, and at the time of boot, reading appearance of the code for the boot process of others in a boot ROM 2 is carried out from a boot ROM 2 by the program performed by the beginning stored in the boot ROM 2, and it is stored by it. Moreover, the code of the boot process stored in main memory 7 is performed by the processor, and loading of the OS (operating system) is carried out, and also various programs, data, etc. are stored.
[0021] Next, it explains, referring to the flow chart shown in drawing 2 about actuation of the 1st example. First, when a system-reset signal is ON, according to the processor optional feature 14, the system bus mediation device 3 outputs the reset signal of the processor control signals S1, S2, and S3 to each of each processors 4, 5, and 6, and continues applying reset to a processor. Consequently, processors 4, 5, and 6 do not operate an instruction fetch etc. at all.
[0022] When a system-reset signal becomes off and a system reset is canceled, the system bus mediation device 3 makes actuation of processors 4, 5, and 6 start by canceling the reset signal of the processor control signals S1, S2, and S3 (off) (step A1).
[0023] Since processors 4, 5, and 6 fetch the instruction of the head address of a boot ROM 2 at this time, system bus mediation device control signal S4 and the system bus acquisition demand signal of S5 and S6 are used, respectively, and a system bus acquisition demand is advanced to the system bus mediation device 3 (step A2).
[0024] The processor to which any the processors by which the system bus mediation device's 3 advanced the system bus acquisition demand being, and a system bus acquisition demand signal have been outputted normally and which can be operated holds any they are to the processor holding register 13 which can be operated (step A3).
[0025] Moreover, out of the processor by which the information that it could operate to the processor holding register 13 which can be operated was held, according to the regulation which specifies priority,

the system bus mediation device 3 chooses the processors (mounting position to the slot of a board in which the processor was carried etc.) which are one with the highest priority, and outputs a system bus licence signal according to the processor optional feature 14. Moreover, the system bus mediation device 3 outputs a reset signal to other processors which were not chosen (step A4).
[0026] Here, it shall be a processor 4 about the processor which inputted the system bus licence signal from the system bus mediation device 3. That is, the processor optional feature 14 uses the system bus licence signal of the processor control signal S1 for a processor 4, gives the licence of a system bus, and is made not to operate an instruction fetch etc. to processors 5 and 6 at all using the reset signal of the processor control signals S2 and S3. That is, in this condition, the processor which operates turns into only a processor 4.
[0027] Since the processor 4 acquired the royalty of a system bus 1, it reads a code through a system bus 1, and performs processing of initialization of initialization of a boot process (i.e., the processor itself), the circumference circuit on a board, etc., initialization of an I/O device, etc. (step A5).
[0028] If a boot process is completed, the processor holding register 13 which can be operated is read, the processor which can operate recognizes any they are, and a processor 4 writes in the suitable value for a status register, a program counter, etc. to the processor which can operate, and it will be changed into the condition that the processor itself was initialized and it will change it into the condition that-izing can be carried out [multiprocessor]. Moreover, the reset signal of the processor control signals S2 and S3 from the system bus mediation device 3 is made to cancel, and actuation of processors 5 and 6 is made to start (steps A6 and A7).
[0029] In the 1st example mentioned above, it explains that a processor with a failure can be eliminated. When there is a processor which has a failure in processors 4, 5, and 6, when a system reset is canceled, a system bus acquisition demand signal cannot be probably taken out to the system bus mediation device 3.
[0030] For this reason, about the processor which was not able to output a system bus acquisition demand signal since it was with obstacles, the information which shows a normal thing is not set as the processor holding register 13 of the system bus mediation device 3 which can be operated.
[0031] Therefore, to the processor which was with obstacles, values, such as a status register of the processor which was with obstacles, and a program counter, will not be written in by one processor which performed the boot process, the reset signal from the system bus mediation device 3 to the processor will not be canceled, and the processor can be eliminated by it.
[0032] Thus, when a reset signal is canceled, a processor with the failure which cannot output a system bus acquisition demand signal to the system bus mediation device 3 is eliminated, and a boot process can be performed by other normal processors. Therefore, a system can be started, if a processor normal otherwise exists even if one highest processor of a priority that should perform a boot process has failure.
[0033] Next, the 2nd example of this invention is explained. Drawing 3 is the block diagram showing the outline configuration of the multiprocessor system concerning the 2nd example of this invention. As shown in drawing 3 , through the system bus 1, it connects mutually and a boot ROM 2, the system bus mediation device 23, the processors 4, 5, and 6 of plurality (this example three), and main memory 7 are constituted. In addition, the same sign is attached about the same component as the multiprocessor system of the 1st example shown in drawing 1 , and explanation is omitted.
[0034] The system bus mediation device 23 arbitrates the use demand to the system bus 1 from processors 4, 5, and 6 etc. It connects with the system bus 1, and also the system bus mediation device 23 performs transmission and reception of a processor control signal and a system bus mediation device control signal among each processors 4, 5, and 6. In addition, the processor control signal includes the reset signal to a processor, and the system bus licence signal from the system bus mediation device 23. Moreover, the system bus mediation device control signal includes the system bus acquisition demand signal over the system bus mediation device 23 from the processor. About the details of the actuation by which each signal is transmitted and received, it mentions later.
[0035] The processor holding register 33 which can be operated, the processor optional feature 34, and

the initialization control flag 35 are formed in the system bus mediation device 23. The processor holding register 33 which can be operated is for holding the information which shows the processor which can operate within a system for every processor, the field of the number of the processors in a system and the same number is prepared, and the field and a processor are matched. The physical address is assigned to the processor holding register 33 which can be operated, and it can write from the exterior (from any of other processors). The processor optional feature 34 sends out the reset signal of a processor control signal for a system bus licence signal only to the processor which can operate to the system bus acquisition demand signal from a processor at a processor according to sending out or a system-reset signal with reference to the information stored in the processor holding register 33 which can be operated. The initialization control flag 35 is used in order to make one processor continue and perform a boot process in two or more processors. The physical address is assigned and the initialization control flag 35 can be written from the exterior (from any of other processors).

[0036] Next, it explains, referring to the flow chart shown in drawing 4 about actuation of the 2nd example. First, when a system-reset signal is ON, according to the processor optional feature 34, the system bus mediation device 23 outputs the reset signal of the processor control signals S1, S2, and S3 to each of each processors 4, 5, and 6, and continues applying reset to a processor. Consequently, processors 4, 5, and 6 do not operate an instruction fetch etc. at all.

[0037] When a system-reset signal becomes off and a system reset is canceled, the system bus mediation device 23 makes actuation of processors 4, 5, and 6 start by canceling the reset signal of the processor control signals S1, S2, and S3 (off) (step B1).

[0038] Since processors 4, 5, and 6 fetch the instruction of the head address of a boot ROM 2 at this time, system bus mediation device control signal S4 and the system bus acquisition demand signal of S5 and S6 are used, respectively, and a system bus acquisition demand is advanced to the system bus mediation device 23 (step B-2).

[0039] To the system bus acquisition demand signal from each processor, the system bus mediation device 23 outputs a system bus licence signal to one processor, and grants the royalty of a system bus 1 (step B3).

[0040] The processor which inputted the system bus licence signal reads the program (code) of a boot ROM 2, and starts activation of a boot process. In a boot process, processing (initialization routine) which initializes the processor itself first is performed (step B4).

[0041] The system bus mediation device 23 gives system bus licence to the processor which outputted the system bus acquisition demand signal one by one. For this reason, each processor performs processing which initializes the processor itself, respectively.

[0042] Consequently, any one of the processors 4, 5, and 6 completes activation of the routine which initializes the processor of the code (program) first stored in the boot ROM 2 itself (step B5). Here, the processor 4 should complete initialization routine first. After the routine which initializes the processor itself is completed, the code which performs the following processings is stored in the boot ROM 2.

[0043] That is, initialization routine is completed, and the processor 4 which completed initialization routine first sets a flag in the field corresponding to the intraprocessor of the processor holding register 33 of the system bus mediation device 23 which can be operated so that a normal thing may be shown (step B6).

[0044] And if the flag does not stand with reference to the initialization control flag 35 as for the processor 4, it sets a flag so that it may be shown that the processor which already completed initialization routine exists (steps B7 and B8). A processor 4 continues activation of codes after it, such as initialization of the circumference circuit on the boot process following own initialization routine of a processor, i.e., a board, etc., and initialization of an I/O device, (step B9).

[0045] On the other hand, processors 5 and 6 set a flag in the field corresponding to the intraprocessor of the processor holding register 33 of the system bus mediation device 23 which can be operated like a processor 4, if initialization routine is completed normally (step B6).

[0046] And since the flag already stands when own initialization routine of a processor is completed to the 2nd shift although processors 5 and 6 refer to the initialization control flag 35 (step B7), they do not

continue and perform initialization routine but perform the meaningless processing performed until it becomes a predetermined condition, i.e., an endless loop, (step B10). For example, in case it is multiprocessor-ized by OS, the dummy instruction (nop instruction) used as the object rewritten by branch instruction is included in the endless loop.

[0047] Since a processor 4 performs a boot process normally, loading of the OS is carried out and it performs it. A processor 4 distinguishes the processor (own initialization of a processor was completed) which can operate with reference to the processor holding register 33 of the system bus mediation device 23 which can be operated based on processing for OS to multiprocessor-ize.

[0048] A processor 4 rewrites the dummy instruction in the endless loop stored in the main memory 7 which the processor 5 and 6 which can operate, i.e., processors, is performing to the branch instruction which branches to the processing which performs multiprocessor processing, takes it out from processing of an endless loop, and makes actuation start substantially.

[0049] In addition, in order to take out from processing of an endless loop to the processor of the 2nd henceforth which completed the routine which initializes the processor itself, when the instruction more than the count of predetermined is executed by the endless loop other than the method of rewriting the above instructions, it branches to the processing which performs multiprocessor processing from an endless loop. In this case, before coming out of an endless loop, the boot process by the processor 4 needs to be completed. Moreover, you may be the other methods.

[0050] In the 2nd example mentioned above, it explains that a processor with a failure can be eliminated. When there is a processor which has a failure in processors 4, 5, and 6, the processor cannot be probably performed to the processing which sets a flag to the processor holding register 33 which can be operated in the field corresponding to intraprocessor, without the ability performing the routine which initializes the processor itself.

[0051] Therefore, since it will not be recognized as the processor which can operate in case it multiprocessor-izes by OS, the processor can be eliminated. Thus, a processor with the failure which cannot perform the routine which initializes the processor itself stored in the boot ROM 2 is eliminated, and a boot process can be performed by other normal processors. Therefore, a system can be started, if a processor normal otherwise exists even if one highest processor of a priority that should perform a boot process has failure.

[0052] Next, the 3rd example of this invention is explained. Drawing 5 is the block diagram showing the outline configuration of the multiprocessor system concerning the 3rd example of this invention. As shown in drawing 5 , through the system bus 1, it connects mutually and a boot ROM 2, the system bus mediation device 43, the processors 4, 5, and 6 of plurality (this example three), and main memory 7 are constituted. In addition, the same sign is attached about the same component as the multiprocessor system of the 1st example shown in drawing 1 , and explanation is omitted.

[0053] The system bus mediation device 43 arbitrates the use demand to the system bus 1 from processors 4, 5, and 6 etc. It connects with the system bus 1, and also the system bus mediation device 43 performs transmission and reception of a processor control signal and a system bus mediation device control signal among each processors 4, 5, and 6. In addition, the processor control signal includes the reset signal to a processor, and the system bus licence signal from the system bus mediation device 43. Moreover, the system bus mediation device control signal includes the system bus acquisition demand signal over the system bus mediation device 43 from the processor. About the details of the actuation by which each signal is transmitted and received, it mentions later.

[0054] The processor holding register 53 which can be operated, the processor control section 54, and the counter 55 are formed in the system bus mediation device 43. The processor holding register 53 which can be operated is for holding the information which shows the processor which can operate within a system for every processor, the field of the number of the processors in a system and the same number is prepared, and the field and a processor are matched. The physical address is assigned to the processor holding register 53 which can be operated, and it can write from the exterior (from any of other processors). The processor control section 54 sends out the reset signal of a processor control signal for a system bus licence signal to the processor which can operate to the system bus acquisition

demand signal from a processor at a processor according to sending out or a system-reset signal. Moreover, the processor control section 54 continues and performs a boot process by the processor which can operate based on the information stored in the processor holding register 53 which can be operated, and the notice from a counter 55. A counter 55 shall measure time amount (for example, 1.5 times to about 2 times) long enough compared with the time amount taken to complete activation of the routine to which it is for measuring predetermined time amount, and a processor initializes the processor in a boot ROM 2 itself after a processor starts a boot process. For example, a counter 55 shall count the number of cycles longer enough than the number of cycles which activation of the routine which is the counter which synchronizes with a system clock and initializes the processor itself takes. A counter 55 is notified to the processor control section 54, when measurement of predetermined time amount is completed.

[0055] Next, it explains, referring to the flow chart shown in drawing 6 about actuation of the 3rd example. First, when a system-reset signal is ON, by the processor control section 54, the system bus mediation device 43 outputs the reset signal of the processor control signals S1, S2, and S3 to each of each processors 4, 5, and 6, and continues applying reset to a processor. Consequently, processors 4, 5, and 6 do not operate an instruction fetch etc. at all.

[0056] When a system-reset signal becomes off and a system reset is canceled, the system bus mediation device 43 makes actuation start by choosing from two or more processors the processors (mounting position to the slot of a board in which the processor was carried etc.) which are one with the highest priority according to the regulation which specifies priority, and canceling a reset signal (OFF). The system bus mediation device 43 keeps a reset signal outputted to other processors (step C1).

[0057] Here, it shall be a processor 4 about the processor by which the reset signal was canceled of the system bus mediation device 43. That is, the processor control section 54 cancels the reset signal of the processor control signal S1 to a processor 4, and gives the licence of a system bus using the system bus licence signal according to a system bus acquisition demand signal, and is made not to operate an instruction fetch etc. to processors 5 and 6 at all using the reset signal of the processor control signals S2 and S3.

[0058] Since the processor 4 acquired the royalty of a system bus 1, it reads a code through a system bus 1, and starts actuation of a boot process. Processing which initializes the processor itself first is performed in a boot process (step C2).

[0059] On the other hand, a counter 55 starts measurement of time amount at the same time actuation of a boot process is started (step C3). The processor control section 54 supervises whether a flag is set as the field of the processor holding register 53 corresponding to the processor which outputted the system bus licence signal, i.e., the processor which is performing own initialization of a boot process of a processor, which can be operated (step C4).

[0060] After ending the routine which initializes the processor itself among the codes in a boot ROM 2 (program), the processor 4 which performs a boot process sets a flag, as a normal thing is shown in the field corresponding to the intraprocessor 4 of the processor holding register 53 which can be operated (step C5).

[0061] If it detects that the flag was set as the field of the processor 4 which is carrying out current actuation, the processor control section 54 will reset a counter 55, and will terminate measurement of time amount (step C6). A processor 4 continues activation of initialization of the circumference circuit on the boot process which follows initialization routine of a processor own [in a boot ROM 2] as it is, i.e., a board, etc., initialization of an I/O device, etc. (step C7).

[0062] If the processor which continues and performs a boot process is obtained, the processor control section 54 will perform a boot process, permitting use of a system bus 1 with a system bus licence signal one by one to other processors (processors 5 and 6) (step C8).

[0063] If the routine which initializes the processor itself is completed, initialization routine will be completed, and the processors 5 and 6 which perform a boot process 2nd henceforth will set a flag in the field corresponding to the intraprocessor of the processor holding register 53 of the system bus mediation device 43 which can be operated so that a normal thing may be shown (step C9).

[0064] Processors 5 and 6 will perform the meaningless processing performed until it becomes a predetermined condition, i.e., an endless loop, if a setup of the flag to the processor holding register 53 which can be operated is completed.
[0065] In case the processors 5 and 6 which perform processing of an endless loop are multiprocessor-ized by OS in which loading was carried out to main memory 7 like the 2nd example mentioned above by the processor 4 which performed the boot process normally, they start [that an instruction is rewritten etc. and] actuation substantially by it.
[0066] On the other hand, when a setup of the flag to the flag 53 of the processor holding register which can be operated is overdue with a certain factor, or when it becomes impossible, a counter 55 completes measurement of predetermined time amount, and notifies that to the processor control section 54 (step C10).
[0067] If the notice from a counter 55 is received, the processor control section 54 will output a reset signal, in order to stop activation of a boot process as what has abnormalities, such as failure, to a working processor (in this case, processor 4) then (step C11).
[0068] Next, the processor control section 54 chooses as the degree of a processor 4 the processor (it considers as a processor 5) which is one with a high priority, and performs processing which initializes a boot process (i.e., the processor itself) like the processor 4 which outputted and (reset discharge) mentioned the system bus licence signal above (steps C1 and C2).
[0069] Under the present circumstances, a counter 55 starts measurement of time amount from an initial state by having changed the processor which performs a boot process (step C3). Hereafter, it processes like the above-mentioned.
[0070] That is, a processor is made to perform a boot process, and when the routine which initializes the processor itself is not able to be completed, the target processor is changed, and it carries out one by one repeatedly until it tries about all the processors in a system.
[0071] Thus, choose one processor which should perform a boot process and the routine which initializes the processor itself is made to try, and if initialization processing is completed normally, a boot process will be continued and performed as it is. And when the routine which initializes the processor itself is not able to be completed in predetermined time amount, the following processor is chosen and a boot process (initialization routine) is tried similarly. Therefore, if the processor which can operate at least one exists in a system, after the routine which initializes the processor itself can be continued, the code in a boot ROM 2 (program) can be performed, and a system can be started.
[0072]
[Effect of the Invention] As explained in full detail above, it becomes possible, since a boot process can be performed by other normal processors and a system can be started, even if the processor which broke down in two or more processors exists according to this invention to constitute a reliable multiprocessor system.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] The block diagram showing the configuration of the multiprocessor system concerning the 1st example of this invention.
[Drawing 2] The flow chart for explaining actuation of the 1st example.
[Drawing 3] The block diagram showing the configuration of the multiprocessor system concerning the 2nd example of this invention.
[Drawing 4] The flow chart for explaining actuation of the 2nd example.
[Drawing 5] The block diagram showing the configuration of the multiprocessor system concerning the 3rd example of this invention.
[Drawing 6] The flow chart for explaining actuation of the 3rd example.
[Description of Notations]
1 [-- A processor, 7 / -- Main memory, S1 S2, S3 / -- A processor control signal, S4 S5, S6 / -- System bus mediation device control signal.] -- A system bus, 2 -- A boot ROM, 3, 23, 43 -- A system bus mediation device, 4, 5, 6

[Translation done.]